利用 BERT 之圍棋 AI 下子預測
BERT for Go Prediction

指導教授：高宏宇 教授 專題成員：蘇晟翔

摘要

圍棋 AI 發展至今，一直是使用 ResNet 相關架構，將棋局轉換成圖片來處理。而近年來 Transformer 因為能處理文字序列成為深度學習的主流，並且我發現儲存圍棋資訊的.sgf 檔案內容為一串 token，這些 token 屬於一個有 361 個字的字典，像是一段句子，可以丟進 BERT 訓練！因此我使用 BERT 訓練圍棋落子選點器(圍棋 AI 的前半部分)，並模仿 Alpha Go，訓練了他使用的 ResNet 模型作為比較標準，分析比較兩者的結果。

研究方法

一、模型說明

1. 都是使用空模型，無預訓練。
2. 做圍棋 AI 前半部分—361 分類器。
3. 根據棋譜第 1~m 步資訊，預測第 m+1 步，做監督式訓練。
4. ResNet：使用 Leela zero 的 ResNet 架構
   - Leela zero 是全球知名圍棋開源專案，實做 Alpha Go 論文
   - 我擷取其中的"落子選點器"部分模型來使用
   - 他也曾經做過以"純人類棋譜"訓練的版本，達到 ELO 2638 分
   - 世界頂尖好手約在 ELO 3600 分
   - 以台灣職業四段俞俐均 2759 分為標準，2638 分約為台灣職業三段
   - https://github.com/leela-zero/leela-zero
5. BERT：使用 Huggingface 的空模型
   - 自訂模型參數。{ vocab_size = 363、type_vocab_size = 7、…}
   - 使用 mean of last hidden states 來分類而非[CLS]

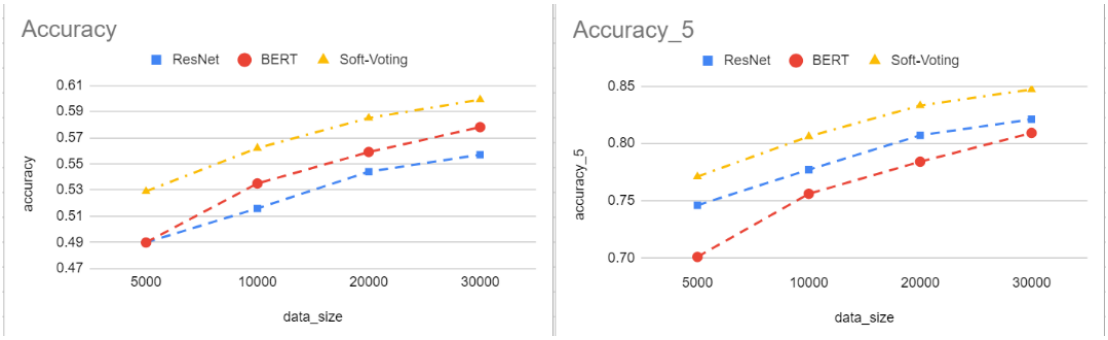
二、資料說明

1. 資料來源:
   - 訓練資料:職業選手大賽棋譜 (http://sinago.com/qipu/new_gibo.asp)
   - 測試資料:野狐軟體九段棋譜 (https://github.com/featurecat/go-dataset)
2. 共同資料處裡:
   - 取每盤棋的前 240 步，因為一般來說這時棋局已經差不多了
   - 因此測資會有[240x(盤數)]筆
3. ResNet 資料處裡:
   - 模仿 Alpha Go 的做法，但簡化內容使其記憶體容量跟 BERT 的一樣
   - 用(1,4,19,19)的圖片表示當前盤面
   - 使用 19x19 的陣列填入 0 或 1 代表各位置
4. BERT 資料處裡:
   - 361 個位置形成 361 個 token，加入"SEP"在結尾，其餘補 0
   - 將落子順序轉化為一個序列
   - 輸入為三個 (1,240)，包含 input_ids、masks、token_type_ids

	BERT	ResNet
資料大小	(1,240)x3，包含 input_ids,mask,token_types	(1,4,19,19)，一張4 channel 的圖片
棋子位置	棋盤上的361個位置對應為數字1~361	使用19x19的陣列填入0或1代表各位置
資料內容	這盤棋過去所有棋子的落子順序	當下棋盤狀況，不包含被吃掉的棋子
資料增加	模仿 mask 的方式加入需要的 embedding	增加 channel 的數量儲存不同資訊

結果與討論

一、Accuracy、Top_5_Accuracy



1. Accuracy 是 BERT > ResNet，顯示 BERT 即使只看到 token，也可以訓練起來。
2. Accuracy 卻是 BERT < ResNet，與 accuracy 相反，後面會再討論原因。
3. 使用 Soft-Voting 方法做 ensemble 對兩種 accuracy 皆有幫助。

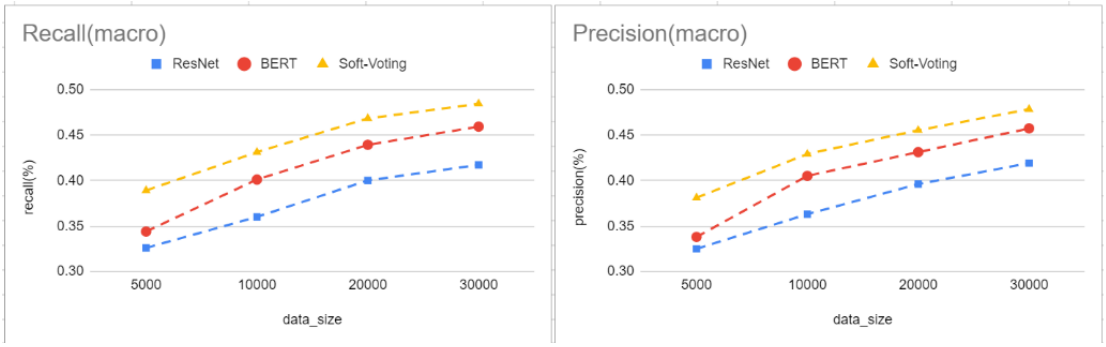
二、不同模型的 ensemble 效果

使用 ResNet(R)跟 BERT(B)各自訓練出兩個模型，進行 ensemble 的結果比較

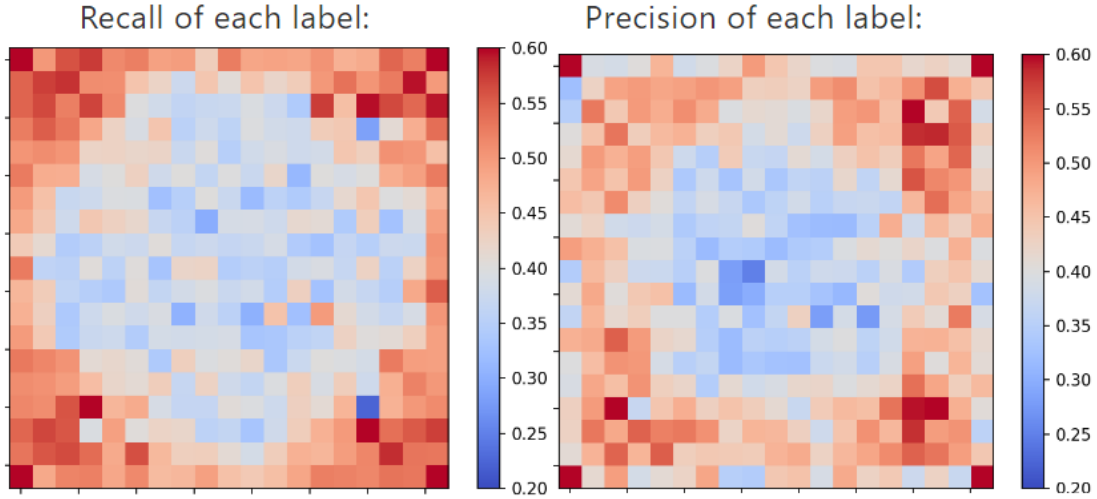
	B1+B2	R1+R2	R1+B2	B1+R2
Both correct	47.9%	45.2%	43.7%	43.8%
Both wrong	38.6%	39.2%	36.2%	36.0%
Others	13.2%	15.6%	**20.1%**	**20.2%**
Soft-voting	56.3%	54.9%	**57.4%**	**57.5%**

1. Others 為兩模型不同時答對或不同時答錯的比率，結果顯示不同種模型明顯擅長不同狀況。
2. 不同種模型做 Soft-Voting 後的 accuracy 也比同種模型高。
3. 顯示兩種模型互補效果不錯。

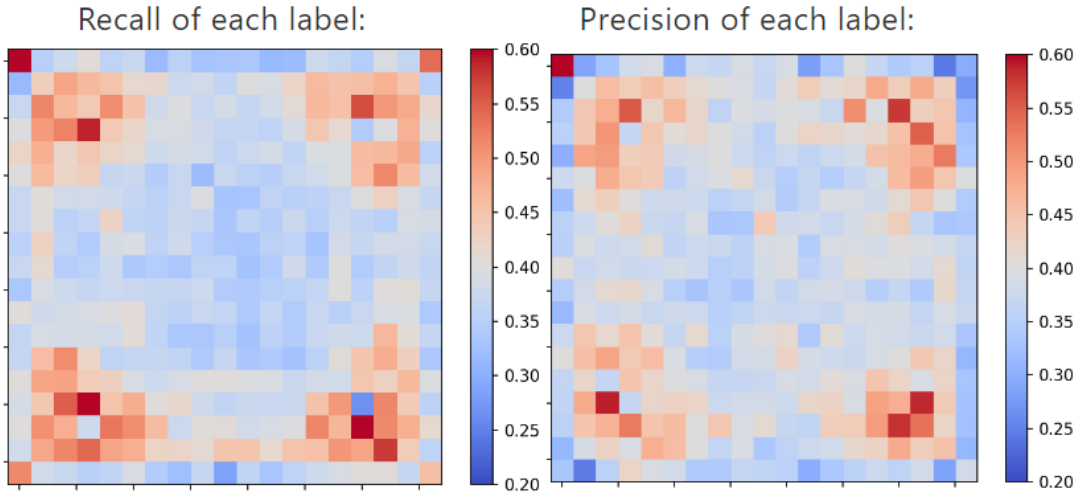
三、Recall / Precision 分析



BERT:



ResNet:



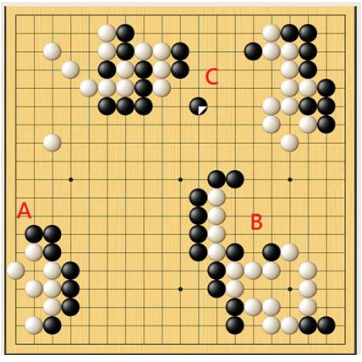
1. 在 micro 下，recall 及 precision 的分數與 accuracy 結果相同，BERT 比 ResNet 高 1.5%。
2. 在 macro 下，BERT 比 ResNet 高 3~4%，差距比 micro 拉大。
3. 顯示 BERT 在一些小類別上做得比 ResNet 好。
4. 比較兩模型的分布圖，BERT 比較擅長最邊緣以及兩個角中間的邊上。

四、不同種模型的訓練時間差異

1. minutes/epoch — **B：R = 7：5**；num_eopchs — **B：R = 4：1** 。
2. BERT 需要的訓練時間比 ResNet 多。
3. 但訓練時觀察到隨著資料量增加，num_eopchs 正逐漸趨向 **B：R = 3：1**。

五、Accuracy、Top_5_Accuracy 結果相反

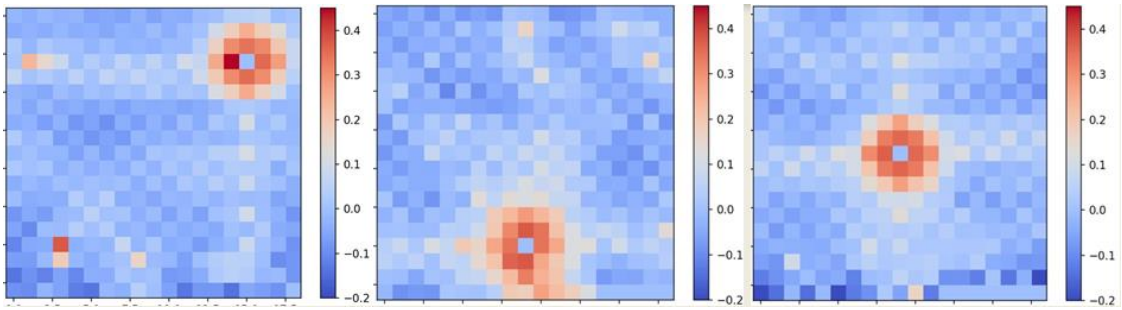
All	B & R	!B & R	B & !R	!B & !R
3.07	2.01	5.91	3.66	6.7



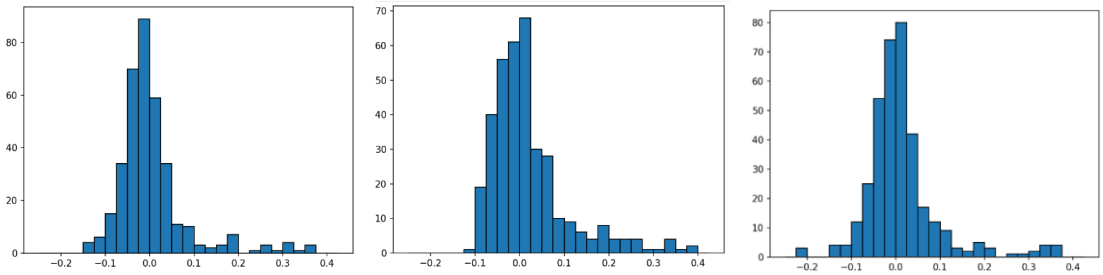
1. 在 top_5_accuracy，「BERT 錯 ResNet 對」的局面，很多是「脫先」，脫先像是一種題型，其特徵是這一手棋距離上一手棋位置較遠，屬於不同「戰場」。
2. 假設要從戰場 A 轉到 B，BERT 的選擇依然集中在 A，導致前五選都判斷錯誤，也就是 BERT 容易在同一戰場持續纏鬥，像是注意力過度集中在最後幾步棋。
3. 在 top_5_accuracy 的條件下，計算各種狀況每一手棋與上一手棋的距離，可以看到 !B & R 比 B & !R 大不少，可見 B 錯 R 對的情況下，脫先出現的比率大概較高，藉此驗證我的觀察。

六、BERT 中各 token 間的 cosine 相似度

觀察各 token 與其他 token 的 cosine 相似度(目標 token 在紅區正中)

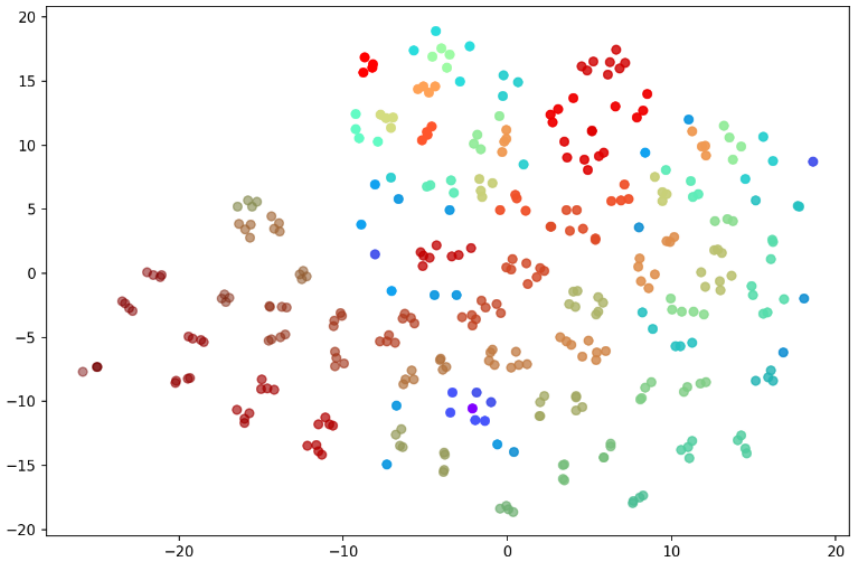


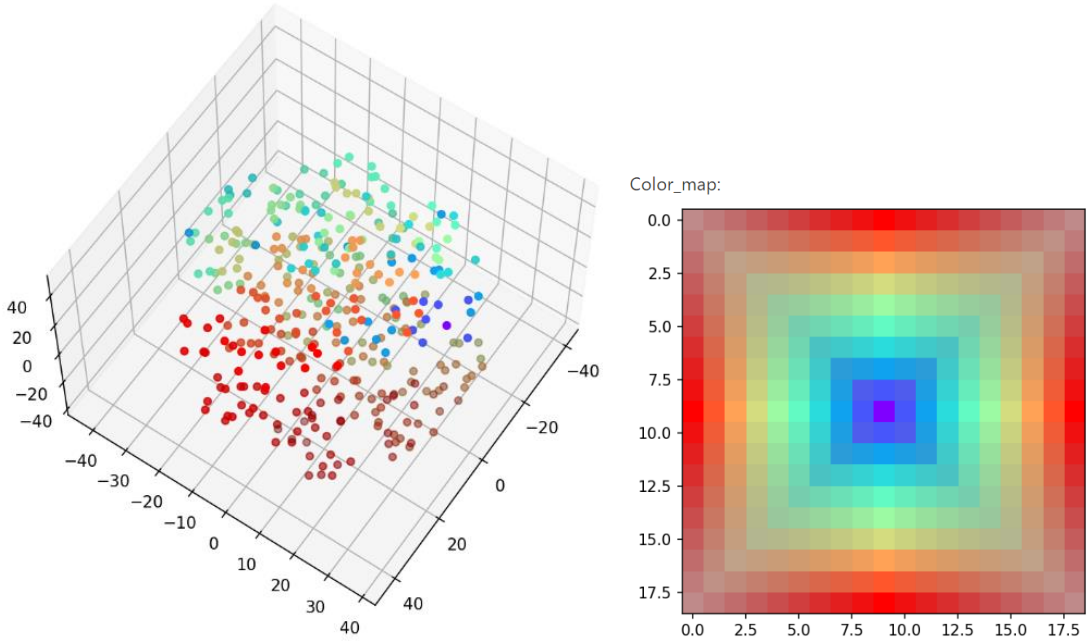
觀察整張相似度的數值大小分布



1. 四角的 token 紅區會出現在其他角落、邊上的 token 紅區會延伸到邊緣、中央的 token 紅區僅集中在自身附近。
2. 根據分布圖，中央的 token 在棋盤上的淺色區較多，邊角的則深色較明顯。
3. 以上 token 間的關係，跟人下棋時的行為策略有些相似:
   - 開局先下各個角落
   - 下在邊上時需要觀察邊緣有無其他子
   - 下在中間時比較需要觀察整個棋盤

七、將 BERT 的 tokens' embedding 可視化來觀察





1. 在二維及三維中都可觀察到，在棋盤上代表顏色相近的 token 其 embedding 結果在圖中的位置也比較近。
2. 在二維圖中可以看到許多 token 常是四個一組，代表棋盤的四個方向。

結論

一、BERT 的優點

1. BERT 的 accuracy 比 ResNet 高，主要因為它在一些小類別的表現比較好。
2. 使用 ensemble 時，用不同種模型做，效率比用同種模型好。

二、BERT 的缺點

1. BERT 的 top_5_accuracy 比較低，目前解決方法只有跟 ResNet 做 ensemble。
2. BERT 的訓練時間比 ResNet 長，不過隨著資料量增加，此問題似乎在改善。

三、BERT 適合圍棋嗎

1. 觀察訓練結果及 token embedding，BERT 確實可以學習圍棋，即使我們只是輸入一串 token 序列，而非棋盤的樣子。
2. 可能可以跟 ResNet 做 ensemble 來使用。
3. 單獨使用 BERT 訓練圍棋前，需要先解決以上兩個缺點，這兩點都很重要。
4. 當資料量夠大時，期待訓練結果或許能產生質變，如同現今的 LLM 一樣。
5. 進一步訓練會需要用到 RL，不過 Transformer 架構已有 RL 模型可參考。
6. 至於實務上的問題，下圍棋需要知道對方前幾步怎麼下的嗎？目前廣泛認為的觀點是，對機器:沒影響；對人類:一定有影響。